Determining Robot Actions For Tasks Requiring Sensor Interaction

John Budenske
Honeywell Systems and Research Center
3660 Technology Drive, Minneapolis Mn., 55418
Maria Gini
Department of Computer Science
University of Minnesota, Minneapolis, Mn. 55455

Abstract

The performance of non-trivial tasks by a mobile robot has been a long term objective of robotic research. One of the major stumbling blocks to this goal is the conversion of the high-level planning goals and commands into the actuator and sensor processing controls. In order for a mobile robot to accomplish a non-trivial task, the task must be described in terms of primitive actions of the robot's actuators. Most non-trivial tasks require the robot to interact with its environment; thus necessitating coordination of sensor processing and actuator control to accomplish the task. Our contention is that the transformation from the high level description of the task to the primitive actions should be performed primarily at execution time, when knowledge about the environment can be obtained through sensors. We propose to produce the detailed plan of primitive actions by using a collection of low-level planning components that contain domain specific knowledge and knowledge about the available sensors, actuators, and sensor/actuator processing. This collection will perform signal and control processing as well as serve as a control interface between an actual mobile robot and a high-level planning system. Previous research has shown the usefulness of high-level planning systems to plan the coordination of activities such to achieve a goal, but none have been fully applied to actual mobile robots due to the complexity of interacting with sensors and actuators. This control interface is currently being implemented on a LABMATE mobile robot connected to a SUN workstation and will be developed such to enable the LABMATE to perform non-trivial, sensor-intensive tasks as specified by a planning system.*

1. Introduction

In order to perform intelligent tasks, a robot needs to interact with its environment through its sensors and actuators. Often information about the environment must be obtained before decision making, scheduling, planning, and verification of high level tasks can proceed. Since the world is large and dynamic, it is impossible to store current information a priori into the robot's internal model of the world, and thus the robot must gather the necessary information through its sensors. The gathering of information could also include the robot's observation of its own interaction with the environment through its actuators.

The intelligent behavior exhibited by the robot in its performance of a task can be depicted or represented as a plan (ordered sets of goals, events, and actions). Initially, a plan consists of a few abstract or high level goals in which information-gathering needs are implicitly understood. During the planning process, the high level goals are transformed into lower level, less abstract goals. Eventually, the implicit information needs of the plan must be identified and explicitly stated, and can be represented as "information gathering goals" within the plan.

In order for any plan to be executed, all higher level goals must eventually be mapped to the robot's sensor and actuator functions. Proper identification and mapping of the information gathering goals require the ability to:

*This research was sponsored in part by the National Science Foundation under grant NSF DMC 8518735.

(1) represent the relationship between high level goals and their implicit information gathering needs (ie. what kind of sensor interaction is needed for what goal),

(2) describe sensors and their ability to satisfy the information gathering needs,

(3) describe (and represent) the translation of informational needs into robot sensor and actuator actions, and

(4) replace the information gathering goals, within the plan, with sensor and actuator manipulations.

The final plan of robot sensor and actuator manipulations must be at such a level of detail so to allow proper execution of the plan by the robot. Our research will address the last three of the above abilities, and only slightly address the first one.

2. Objectives

The objective of this research is to design and develop a sensor and actuator control interface to the LAB-MATE mobile robot, resident in the Artificial Intelligence Laboratory of the University of Minnesota, Computer Science Department. This control interface has three main functions:

(1) intelligently process sensor information for use by a high-level planning and reasoning system (here on referred to as the Planning system);

(2) intelligently control the robot's sensors and actuators via the combination of directive control (from commands/plans sent to it by the Planning system) and reactive control (from reasoning about sensor data collected from the robot's environment), and

(3) provide a "knowledgeable" interface between the robot's control levels and the Planning system.

The first function of the control interface is to process the sensor data for use by a Planning system, as well as by any other sub-system of the robot. A Planning system is a symbolic level reasoning system, such as an expert system or automated planning/scheduling system. The control interface should allow any of the robot's sub-systems to access varying levels of raw and processed data from the robot's sensors. Thus both raw data and abstracted data will be available to the Planning system for reasoning.

The second function of the control interface is to allow the robot to possess a reactive behavior to its environment and yet be effectively controlled by a Planning system (here on referred to as directive behavior). A reactive behavior is the ability to conform the desired actions of the robot to correspond to the current state of the robot's environment. It requires the collection of sensor data, and the calculation of the responding actuator commands. This behavior is often cited along with survivalistic characteristics such as obstacle avoidance, and getting out of the way of big meteorites. Though reactive behavior is necessary for effective interaction with the robot's environment, directive behavior is necessary for the robot to derive the set of actions and intermediate goals which solves a high-level task. The control interface will combine both behaviors such that high-level tasks can be accomplished.

The final function is to provide a "knowledgeable" interface between the robot's control functions and the Planning system. The interface should contain within itself information on the functionality, implementation, status, etc., of the available commands to the Planning system, and this information should be accessible to the Planning system. The Planning system, then, must possess the ability to access the information, reason about it, and utilize it to interact with the robot's sensors and actuators.

3. Related Work

The problem of programming a mobile robot to perform various non-trivial and sensor-intensive tasks is not a new problem within the robotics domain. This problem has been attacked from four different directions of research over the past decade. The four are: automated planning for robots, low-level feedback-control loops, subsumption architecture, and hierarchical and distributed architectures. Each approach has definite benefits and drawbacks.

3.1. Automated Planning and Navigation for Robots

There has been a great deal of research done within the domain of automated planning and navigation, especially on domain-independent planning systems. [10,22,24,31,33] The majority of this research was aimed at creating general problem solvers which can be adapted to fit most any domain. Most of these general problem solvers are based on the same classical constructs of goal-reduction, conflict-detection/resolution, and constraint management. These classical planning systems are designed to derive a set of partially ordered primitive actions which will transform an initial world state to a given final state. Some of the classical planning research has been adapted for problems within the mobile robot domain [23,30] and there has been simulated robots running with classically-based planning systems [7,19] controlling them. Most of this research was aimed at the problems of high level task planning and did not address any of the problems of sensor processing, interaction and control.

The Stanford Cart program, [20] the HILARE robot, [6] and research by Moravec and Elfes [21] concentrated on the complexities of sensor processing, sensor data uncertainty, and actuator control within the domain of robot navigation. All attempted to solve problems in robot position uncertainty, obstacle detection/avoidance, and piloting the robot from position to position. Due to poor actuator controllers, large error propagations, long response times, and/or very weak problem solving algorithms, none of these projects attempted any non-trivial, sensor-intensive problems beyond that of simple obstacle avoidance through path planning. Kuipers and Levitt [35] both implemented navigation planning systems for different simulated robots. Both systems were concerned with how to represent space and reason about it, but neither addressed problems in sensor controls, noisy data processing and problems due to a dynamic world.

More recently, Wilkins, and Drummond [9,32] each have made specifications within their plan representations for the use of sensory data. These mechanisms assumed the availability of highly processed sensor data and did not address data processing needs, derivability, uncertainty management, and error recovery. Firby [11] has implemented a planning system to perform reactive planning within a simulation. The system has operators to determine when to use sensors, but does not have constructs to deal with errors, sensor controls and parameters, and inaccurate or noisy sensor data. Finally, Georgeff [12] has implemented a planning/control system initially in simulation and then on the SRI FLAKEY mobile robot where it exhibited 2 main problems: 1) poor response time to sensor input due to delays of interaction between the symbolic level control and the lower-level robot and sensor control; and 2) problems of control and system development due to the normal errors which embody sensor processing as well as sensor data.

Throughout most of the automated planning research for robots, there has been a great deal of simplifying assumptions on the amount of control and sensor processing and interfacing required to control an actual robot. These systems performed well in simulations based on these simplifying assumptions, but never progressed to successful implementation on actual mobile robots. Their assumptions on the management, control and interactions with sensors, sensor data and actuators were far too basic to their design, and thus limited their ability to successfully implement their designs on actual mobile robots.

3.2. Feedback-Control Loops

The research on automated planning for robots was a general problem solving approach to the robot control problem. In contrast, the next direction of research focused on narrowly defined problems where domain specific

knowledge is highly applicable. The portion of this research which is best addressed towards the problem of accomplishing high-level tasks for mobile (or any) robots, has been the work in sensor-actuator feedback-control loops. Here a great deal of research has been performed to develop many different types of systems which continuously transforms a sensor input signal into an actuator control signal, controlling the actuator to accomplish a (usually very low-level) sensor-intensive task (ie. mathematically transform or through look up tables). For example, a number of sensor feedback-control loop systems have been developed for a robot arm to insert a peg into a hole based on compliant motion. [8,13,16,17,34] Compliant motion is a specification of robot motion modification in response to forces generated during the robot motion which enables the robot to carry out a task in the presence of significant sensing and control errors.

There have been many similar successful developments of sensor-actuator feedback control loops which performed single, narrowly defined, low-level, sensor-intensive tasks with actual robots. Such systems are quite sensitive to details of geometry and to error characteristics and must therefore be constructed anew for each task. Though the task complexity has increased in terms of sensor signals and actuator control, the flexibility of the control is still limited to parameterized, direct execution, with no ability to plan or coordinate multiple actions. Higher-level tasks which require flexible control for multiple actions and multiple goals have not been attempted. Also, there have been no attempts at interfacing any feedback-control loops to a Planning system such to enable more flexible control in performing higher-level tasks.

3.3. Subsumption Architecture

The subsumption architecture [5] is a layered approach to building robust sensor-actuator feedback control systems. Mobile robot control is transformed into a problem based on parallel task achieving behaviors. The key idea is that layers of a control system can run in parallel to each other. Each individual layer correspondence to a level of behavioral competence. The next higher level of overall competence (ie. improved intelligence) or enhanced capability can be obtained by adding a new layer to an existing group of layers. The new layer will run in parallel with the other layers, and will interact with the lower layers through excitation of their inputs and inhibition of their outputs. Basically, the higher layer examines the data flowing through the lower layer, and injects data into the layer, suppressing the normal flow. The lower layer runs the same regardless of the existence of the higher layer. Thus, each layer can be frozen after it is thoroughly debugged. Each layer is implemented within a set of small processors, each processor running a finite state machine, and communicating to other processors across single bit data paths.

The main emphasis of the subsumption architecture is the levels of parallel behaviors, with each behavior performing sensor-actuator feedback control tasks. Through this implemented architecture, the robot has been able to wander through an environment, avoid obstacles, search for doorways, and travel through them. Though the distributed method of control for the robot allows for many behaviors to run in parallel (thus performing many sensor-intensive tasks at once), it also disallows the ability to centrally control the robot to achieve planned goals such as are in high-level tasks. There is no ability for a reasoning system to interact with the robot, controlling it to perform high-level tasks (ie. no high-level Planning system interface).

3.4. Hierarchical and Distributed Architecture

The forth approach concentrated on developing specialized architectures for dealing with complexity of processing and data flow. Albus [1] is implementing a highly-synchronized, hierarchy of computational modules to control sensors and process sensor data as it flows though the structure. The longest planning horizons exist in the top computational module where the highest level decisions are carried out as well. At each level, goals and plans are generated, synchronized, and passed onto the appropriate submodule where they are again decomposed into subgoals, and passed on further down the hierarchy. Decomposition is based on processed input data from the sensors, information on the current state of the control hierarchy, and predictions generated by higher level modules. The system is a highly-synchronized, static, configuration of routines and subroutines.

Another hierarchical architecture enlists a declarative description of the sensors combined with procedural definitions of sensor control. This research on "Logical Sensors" [15,2] consists of logical/abstracted views of sensors and sensor processing, much like how logical I/O is used to insulate the user from the differences of I/O devices and computer operating systems. It is similar to a distributed variation of the hierarchical control approach, only it is more net-like than tree-like, and it allows more flexibility in the inter-module control structure. It provides a coherent and efficient data/control interface for acquiring of information from different sensors types.

Finally, one of the more referenced mobile robot architectures is the "blackboard", whose function is to maintain the consistency of sensor data as it is being processed, and manage the sensor control based on the informational needs of other system processes. [29,3,4,14,18] These systems have been used extensively on the DARPA ALV and other similar projects, where they have controlled mobile robots through road following, and obstacle avoidance tasks. Blackboard systems usually require that all sensor data be processed and converted into a single logic/symbolic-based format. This can hamper control processes which must first convert the data from raw sensor to numerical and then to symbolic; then reason about the resulting commands. Then the symbolic commands must be converted to low level control commands before executing them. Unlike the very quick feedback control loop approach, this approach can become very slow, limiting the robot's task performance for even the simplest of tasks.

4. Proposed Work

4.1. Rationale for the Approach and Designs

Past research has shown that the success of a robot accomplishing a task in any unstructured environment highly depends on the ability of the robot to correctly sense its environment and correctly use the information which it senses. All of the research surveyed which initially conducted the experiments in simulation and then attempted to implement in the real world have underestimated the problems with processing sensor data, controlling the sensors, and interacting with the real world such that their systems were unable to effectively control the robot.

Previous research in controlling a robot within unstructured environments have produced only limited success in achieving narrowly defined tasks (ie. road following). The low level feedback control loops may provide speedy response to sensory input, but they only can perform singly-defined tasks, and methods of combining them into goal achieving behaviors are needed. Brook's research is a step towards combining sensor-feedback control loops into intelligent, reactive behaviors, but it does not attempt to solve the problems of allowing goal directed behavior along with the reactive behavior.

One problem with the blackboard-like approaches is that they have been built from the point of view of the reasoning and planning systems. Little emphasis is placed on the needs, capabilities and limitations of the sensor and control systems. Emphasis on these needs, capabilities, and limitations is important in the design of the reasoning and planning system because in order to achieve a given task, reasoning and planning must interact exclusively with sensors and actuator controls. These reasoning systems do not contain sufficient knowledge to reason about the use of the sensors, and are thus unable to fully utilize the sensor to improve the task performance.

A second problem with complex structures and methods is the reactive delay which they exhibit. The complexity of the system causes a very slow response time, and thus not only is the main task performed very slowly, but the ability of the robot to react to unexpected events is extremely slow. A commonly used example of this problem is a robot sitting in the middle of the street, slowly reasoning about which way to move, and being hit by a truck before deciding. Slow reaction time to hazardous and dangerous events is unacceptable for any robot within an unstructured environment.

Collectively, previous research in mobile robots performing tasks in unstructured environments have not emphasized one of three important problems:

(1) the problems of sensors, sensor interaction, and sensor processing,

(2) the need to intelligently control so to accomplish high level tasks, and

(3) the need to have the robot react quickly to its environment (often called reactive behavior).

To accomplish tasks in unstructured environments requires proper interaction with the sensors. This requires intelligent management, representation, interaction and utilization of sensors and sensory data. The problems with sensors can not be ignored if success is expected.

4.2. Specific Aims of the Approach and Design

This research is aimed at overcoming the problems not emphasized by previous work. The approach is to:

(1) develop a control interface system between the robot's sensors/actuators and the Planning system's such to logically abstract the functionality of the robotic sensor/actuator processing and control from the implementation;

(2) build into the interface the capabilities to a) process and reason over sensor data so to convert it into a form usable by a Planning system, b) control sensors and actuators such to drive the robot in both directive and reactive behavior modes, c) allow reasoning, analysis, and recovery over errors occurring during the performance of tasks, and d) allow both a Planning System and the interface components to reason over the use of the interface's functionality; and

(3) test the control interface by a) adapting an available Planning system to the interface such to control the robot to successfully accomplish a high level task; and b) perform experiments on an actual robot and sensors such to attack the problems of real-time sensor control and sensor data processing;

4.3. Design of the Logical Sensor and Actuator System (LSAS)

We are currently implementing a control interface, called the Logical Sensor and Actuator System (LSAS), between the sensor and actuator drivers on the LABMATE robot and a Planning System. This control interface consists of a collection of low-level planning components that contain domain specific knowledge on the accomplishment of high-level Planning goals. These components also contain knowledge on the availability and use of sensors, actuators, and sensor/actuator processing.

The basic component-types of the LSAS are the Logical Sensor, the Sensor Driver, and the Actuator Driver. Logical Sensors are abstract views of robot sensor hardware combined with sensor processing software, which can be extended to include actuator hardware and processing such to produce sensor-actuator feedback control loops. Simple Logical Sensors will sense the environment, process the data as determined by their main functional purpose, and return an output signal back to all requesting processes. This signal can be as complex as a multi-image signal to as simple as a symbolic label. The sensing of the environment can be directly through the sensor (Sensor Driver), or through other Logical Sensors. Thus Logical Sensors can be built hierarchically to produce varying levels of processed sensor data.

Sensor Drivers and Actuator Drivers are the lowest level software interfaces to the actual sensor and actuator hardware. These entities will consist of many of the properties of Logical Sensors, but differ in that there is a one-to-one correspondence between Sensor/Actuator Drivers and the actual hardware components they drive (multiple Logical Sensors which perform the same task can exist to provide flexibility). Sensor Drivers interface directly to the sensor hardware, and control the hardware parameters in accordance to the control and data request commands sent to it by Logical Sensors. Likewise Actuator Drivers interface directly to the robot's actuator drivers and control them in accordance to the control commands sent to it by Logical Sensors.

Logical Sensors are created from the combination of Sensor Drivers, Actuator Drivers, and other Logical Sensors, thus producing a hierarchical structure of sensor-actuator processing and control functionality. For example, a Logical Sensor for searching out thermostat locations could consist of Logical Sensors for: avoiding obstacles, wandering around the environment, going through doorways, and recognizing thermostats. Each of these Logical Sensors provides a functional level of sensor (and/or) actuator processing and control, and each consists of additional sub-sensors, which also provides a sub-level of functionality.

Externally, the Logical Sensor is an abstract view of sensor-actuator functionality. Internally, Logical Sensors contain a main function, a standard set of interaction functions, and a standard set of sensor-facts. All Logical Sensors have the same base set of standard interaction functions and sensor-facts. Interaction functions include status checks on the hardware, hardware initialization routines, and control parameter manipulations. Sensor-facts include information such as the sensor's name, parameter list and types, output data types, possible side effects, power usage estimates, and a list of goals which it can aid in achieving. Each Logical Sensor subtype can add additional interaction functions and sensor-facts if needed. The complete set of functions and sensor-facts will constitute the knowledgeable interface component of the LSAS. Thus, the interface contains knowledge on how to use the Logical Sensor (parameter information); what purpose to use the Logical Sensor (goals it could achieve); what conditions will the Logical Sensor perform well under; and even how to test the Logical Sensor.

Each Logical Sensor also contains a main function which intelligently performs the sensing/actuator processing for which the Logical Sensor exists. This function can vary from performing very simple signal processing to performing complex reasoning over its current situation and sending actuator commands such to accomplish a task. For example, the main function for a simple Logical Sensor which monitors a force sensor until a threshold is reached and then signals another Logical Sensor will be one which interacts with the sensor, compares the force to the threshold, and signals upon reaching the threshold. In comparison, a more complex Logical Sensor which determines the distance to an object directly in front of the robot may first reason about the the differing characteristics of the available Logical Sensors for measuring distances (ie. sonar, infra-red, visible camera, actual distance traversed by robot's wheels,...) vs. the characteristics of the current situation, and select the most appropriate Logical Sensor. Then, it will monitor the execution of the selected Logical Sensor for possible errors. If there is an error, it will reason about its cause, and take appropriate action (which includes selecting a different Logical Sensor). The Logical Sensors which perform complex reasoning can be viewed as micro-planning systems with very small domains (ie. the domain of deriving distances), in which planning/reasoning is performed only to determine the next one or two robot actions. This type of planning is necessary due to the dynamic nature of the world, and will only work for accomplishing goals which are a few actions from success (thus the need for the Planning system to perform long-term planning and coordination of Logical Sensors).

5. Example

The intent of this research is to automatically determine mobile robot actions which will accomplish high-level sensor-intensive tasks. In order for a robot to be useful, it must be able to accomplish tasks which require interaction with its environment. Of course, interaction within unstructured environments is highly desirable, and any task requiring such interaction must be sensor-intensive. This means that a vast majority of the robot's actions require the use of contemporaneous sensor data to assure successful execution.

An example of a non-trivial, sensor-intensive task is the reconnaissance mission. One of the goals of the reconnaissance mission plan would be a follow-recon-path goal. Within this goal, the robot will enter into a known territory (previously charted, ie. by satellite), search for new objects in the environment, and attempt to record data on the object without disrupting the object's existence within the environment. This example assumes either an indoor environment (ie inside a space station), or no problems with terrain locomotion, (wheel slippage, etc.), and environmental hardening (ie. protection against the elements). These problems, though of concern in the field of robotics, are not the emphasis of this research, and thus simplifying assumptions are used.

In this example the robot is given a partially-ordered plan from the Planning system (ie. much like a plan from NOAH [22]), where one of the goals in the plan is follow-recon-path. Each of the plan's goals are given to the control interface one at a time. This example will follow the execution of the follow-recon-path goal as it is given to the control interface. The first step of the control interface in executing the follow-recon-path goal is to break it down into two operations: follow-given-path and search-for-new-object. This decomposition must take place within the control interface for two reasons. First, both operations must be executed within the control interface due to their sensor-intensive nature and the dynamic nature of the environment. Second, though each of these operations achieves a subgoal of the given goal, they must be performed in parallel in order for their combination to achieve the entire goal. Forcing the transition from Planning system to control interface to occur before such goals exist within the plan will simplify the overall process (ie. current planning research is still struggling with reasoning about parallel execution of goals).

In performing the follow-given-path operation, the robot would move through its environment following a path defined with the given goal. As the robot is following this path, it will be performing the search-for-new-object operation in parallel. This operation will consist of the robot comparing the objects found in its environment with those recorded on the given chart (or from the last time the robot was along that path). Note that a third operation of recording a map of its environment or updating that map could also be performed in parallel. Upon discovering a new (or interesting) object in its environment, the robot will then suspend the current two operations, and start a new operation which will record information on the new object. The criteria for "new or interesting" are not important at this point, but could be determined via various sensors such as magnetic fields, infra-red, etc. or via the robot performing comparisons between the current environment and an a priori chart. The objective of the new operation, record-data-on-object, is to use various sensors to record data on the newly discovered object. The recorded data can be analyzed at a later time (ie. transmitted back to a base station for more in depth analysis). A second objective of this operation is to not get too close to the new object such to disturb, disrupt, or be in danger from it. First, the robot will select two to four vantage points to collect information on the object. Interaction with the higher-level Planning system may be required to partially plan ahead as to which vantage-points to visit first as well as to predict which data-recording parameters would be necessary. The robot's Planning system only partially plans, because of world dynamics rendering complete planning untractable; thus, the robot only plans that which is necessary such to detect possible unforeseen interactions (ie. wasting time by poor selection of vantage-points).

The selection of vantage-points, the number of them, and the ordering of them will depend on the location of the new object relative to the robot, the actual existence of "good" vantage points (or any vantage points), limits of the recording devices, desired distance to maintain from the object, etc. This Planning system may need to interact with the robot's sensors such to collect this data. As the plan is being derived, its execution can begin. The robot will proceed to the first vantage point and record data on the object. If the object moves, the robot will re-select (replan) vantage-points and attempt to continue recording data on the object. If the object approaches the robot, the robot will react, and move away, keeping the desired distance. Continued aggression by the object will result in the robot aborting the data recording attempt. Upon completion or abortion of the data recording operation, the robot will return to the point of suspension, possibly requiring interaction with the Planning system, and continue on with its initial two operations (follow-given-path and search-for-new-object).

This example exhibits many characteristics of non-trivial, sensor-intensive tasks. Examples are:
1) The robot must record information on its environment for determining interesting objects.

2) The robot must perform simple navigation tasks to follow a given path and avoid obstacles.

3) The robot is able to perform parallel operations, each achieving separate subgoals of a given goal.

4) The robot is able to suspend and resume operations as dictated by by control and by input sensor data.

5) Upon finding a new object, the robot is able to determine positions for data gathering, partially plan the task of the goal of data gathering, and then accomplish that goal through execution.

6) The robot is able to react to unforeseen occurrences, and replan its actions such to achieve its goal.

This enumeration is not a complete set of characteristics for any non-trivial, sensor-intensive task, but serves as an illustration of what constitutes such a task. It is important to note the amount of sensor interaction which must occur in order for the robot to accomplish the task. The acknowledgment of this high amount of sensor interaction is the motivation of this research.

6. Summary

Mobile robot primitive actions which accomplish non-trivial, sensor-intensive tasks can be determined through the proper use and representation of the sensors and sensor data provided by the LSAS architecture. The LSAS is designed to be an control interface between high-level planning systems, and the sensor and robot hardware through hierarchical layers of sensor and actuator processing and control. Until recently, previous research only emphasized purely reactive systems (which are unable to plan, reason, and control such to achieve given goals), or purely goal directive systems (which are unable to react to unforeseen situations and threats). Within the LSAS, both directive and reactive behavior are possible through multiple parallel Logical Sensors and the combination of their outputs. Previous research which incorporated both behaviors primarily used blackboard approaches, and perform extensive amounts of processing in order to accomplish trivial tasks, thus computational time delays reduce the effectiveness of sensor response. Effective reactive behavior will be achieved in the LSAS by minimizing sensory input response time via the use of feedback control loops. Finally, one feature which many other approaches do not possess is the inclusion of a knowledgeable interface between the sensors and the Planning system. Thus, the Planning system can reason about the which, where, when, and how of using the sensors as it plans future moves.

The LSAS design addresses the need for intelligent processing of sensor data, combining directive and reactive control, and providing a knowledgeable interface to the sensors and actuators. The development of the LSAS will provide greater insight to the use of sensors for accomplishing intelligent behavior within mobile robots, and will provide further research topics in intelligent robot control, sensor interaction and control, multi-sensor fusion for the accomplishment of tasks, sensor utilization to improve task performance, and improved robot-environment interaction through error recovery and reactive behavior techniques.

7. References

1. Albus, J. S., C. R. Mclean, A. J. Barbera, and M. L. Fitzgerald, "Hierarchical Control for Robots and Teleoperators," *Proccedings of the IEEE Workshop on Intelligent Control*, pp. 39-49, 1985.
2. Allen, Peter K., "A Framework for Implementing Multi-Sensor Robotic Tasks," *Procceedings of the DARPA Image Understanding Workshop*, pp. 392-398,1987.
3. Almand, Bonni, "Sensor Information Fusion and a Robot Decisionmaking Methodology," Proceedings of the SPIE Intelligent Robots and Computer Vision vol. 579, pp. 436-441, 1985.
4. Arkin, Ronald C., Edward M. Riseman, and Allan R. Hanson, "AuRA: An Architecture for Vision-Based Robot Navigation," *Procceedings of the DARPA Image Understanding Workshop*, 1987.
5. Brooks, R. A., "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 1, pp. 14-23, 1986.
6. Chatila, Raja and Jean-Paul Laumond, "Position Referencing and Consistent World Modeling for Mobile Robots," *IEEE International Conference on Robotics and Automation*, pp. 138-145, 1985.
7. Dean, Thomas L., R. James Firby, and David Miller, "The FORBIN Paper," YaleU/CSD/RR 550, Yale University, 1987.
8. Drake, S. H., "Using Compliance in Lieu of Sensory Feedback for Automatic Assembly", Department of Mechanical Engineering, Massachusetts Institute of Technology, 1977. Ph. D. Thesis
9. Drummond, Mark, Ken Currie, and Austin Tate, "Contingent Plan Structures for Spacecraft," JPL Workshop, p. 9, 1987.
10. Fikes, Richard E. and Nils J. Nilsson, "STRIPS: The Application of Theorem Proving to Problem Solving," *Artificial Intelligence*, vol. 2, pp. 189-208, 1971.

11. Firby, R. James, "Reactive Execution as a Basis for Strategic Planning," *AAAI Workshop on Planning for Autonomous Mobile Robots*, 1987.
12. Georgeff, M.P., A.L. Lansky, and M. Schoppers, "Reasoning and Planning in Dynamic Domains: An Experiment With a Mobile Robot," SRI Artificial Intelligents Center Technical Note 380, Artificial Intelligents Center, SRI International, Menlo Park, Califomia, 1987.
13. Gottschlich, S.N. and A.C. Kak, "A Dynamic Approach to High-Precision Parts Mating," IEEE International Conference on Robotics and Automation, pp. 1248-1253, Computer Society Press, Philadelphia, Pa., 1988.
14. Harmon, Scott Y., "The Ground Surveillance Robot (GRS): An Autonomous Vehicle Designed to Transit Unknown Terrain," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3, pp. 266-279, 1987.
15. Henderson, Tom, Chuck Hansen, and Bir Bhanu, "The Specification of Distributed Sensing and Control," *Journal of Robotic Systems*, vol. 2, no. 4, pp. 387-396, John Wiley & Sons, Inc., 1985.
16. Lozano-Perez, T., M. T. Mason, and R. H. Taylor, "Automatic Synthesis of Fine-Motion Strategies for Robots," *The International Journal of Robotics Research*, vol. 3, no. 1, pp. 3-24, 1984.
17. McCallion, H. and P. C. Wong, "Some Thoughts on the Automatic Assembly of a Peg and a Hole," *Industrial Automation*, vol. 2, no. 4, pp. 141-146, 1975.
18. Meystel, A., "Planning in a Hierarchical Nested Autonomous Control System," *Proceedings of the SPIE Conference on Mobile Robots*, vol. 727, pp. 42-76, Cambridge, Mass, 1986.
19. Miller, David, "Planning by Search Through Simulation," YALEU/CSD/RR #423, 1985, Ph.D. Thesis.
20. Moravec, Hans P., "The Stanford Cart and the CMU Rover," *Proceeding of the IEEE*, vol. 71, no. 7, pp. 872-884, 1983.
21. Moravec, Hans P. and A. Elfes, "High Resolution Maps from Wide Angle Sonar," International Conference on Robotics and Automation, pp. 116-121, 1985.
22. Sacerdoti, E., *A Structure for Plans and Behavior*, American Elsevier Publ. Company, 1977.
23. Sacerdoti, E., "Plan Generation and execution for robotics," Tech Note 209, SRI, 1980.
24. Tate, Austin, "A Review of Knowledge-Based Planning Techniques," AIAI-TR-9, p. 17, A.I. Applications Institute, University of Edinburgh, 1985.
29. Thorpe, Charles and Takeo Kanada, "1986 Year End Report for Road Following at Carnegie Mellon," CMU-RI-TR-87-11, p. 60, Department of Computer Science, Carnegie-Mellon University, 1987.
30. Vere, Steven A., "Planning in Time: Windows and Durations for Activities and Goals," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 3, pp. 246-267, 1983.
31. Wilkins, D., "Domain independent planning: representations and plan generation," *Artificial Intelligence*, vol. 22, pp. 269-301 , 1984.
32. Wilkins, David, "Recovering from Execution Errors in SIPE," *Computational Intelligence* vol. 1, no. 1, pp. 33-45, 1985.
33. Wilkins, David, Kurt Konolige, Stan Rosenschein, et al., "Research on Planning at SRI International," SRI International, 1987.
34. Xiao, Jing and Richard A Volz, "Design and Motion Constraints of Part-Mating Planning in Presence of Uncertainties," *IEEE International Conference on Robotics and Automation*, pp. 1260-1268, Computer Society Press, Philadelphia, Pa., 1988.
35. Kuipers, Benjamin J., and Tod S. Levitt, "Navigation and Mapping in Large Scale Space" *AI Magazine* vol. 9, no. 2, pp. 25-42, Summer 1988.

NASA LANGLEY RESEARCH CENTER